Priority-based Congestion Control in Wireless Sensor Networks

Conference Paper · July 2006

DOI: 10.1109/SUTC.2006.108 · Source: IEEE Xplore

5 authors, including:

Chonggang Wang
Institute of Electrical and Electronics Engineers

Khosrow Sohraby
Bayer

Bo Li
The Hong Kong University of Science and Technology

Yueming Hu
South China University of Technology

Priority-based Congestion Control in Wireless Sensor Networks

Chonggang Wang[1], Kazem Sohraby[1], Victor Lawrence[2], Bo Li[3], Yueming Hu[4]

[1]*Dept. Of Elec. Engi., University of Arkansas, Fayetteville, AR 72701, USA*
[2]*Stevens Institute Technology, Hoboken, NJ 07030, USA*
[3]*Dept. of Computer Science, Hong Kong Univ. of Science & Tech., Hong Kong, P.R.China*
[4]*South China Agriculture University, Guangzhou, China*
E-mail:{cgwang, sohraby}@uark.edu, vlawrence@stevens.edu, bli@cs.ust.hk

Abstract

In wireless sensor networks (WSNs), congestion occurs, for example, when nodes are densely distributed, and/or the application produces high flow rate near the sink due to the convergent nature of upstream traffic. Congestion may cause packet loss, which in turn lowers throughput and wastes energy. Therefore congestion in WSNs needs to be controlled for high energy-efficiency, to prolong system lifetime, improve fairness, and improve quality of service (QoS) in terms of throughput (or link utilization) and packet loss ratio along with the packet delay. This paper proposes a node priority-based congestion control protocol (PCCP) for wireless sensor networks. In PCCP, node priority index is introduced to reflect the importance of each node. PCCP uses packet inter-arrival time along with packet service time to measure a parameter defined as congestion degree and furthermore imposes hop-by-hop control based on the measured congestion degree as well as the node priority index. PCCP controls congestion faster and more energy-efficienty than other known techniques.

1. Introduction

Wireless sensor networks (WSNs) [1] have attracted tremendous attention in both academia and industry in recent years. A WSN consists of one or more sinks and perhaps tens or thousands of sensor nodes scattered in an area. The upstream traffic from sensor nodes to the sink is many-to-one multi-hop convergent. A WSN consists of one or more sinks and perhaps tens or thousands of sensor nodes scattered in an area. The upstream traffic can be classified into four delivery models: event-based, continuous, query-based, and hybrid delivery. Due to the convergent nature of upstream traffic, congestion more probably appears in the upstream direction. Congestion that can leads to packet losses and increased transmission latency has direct impact on energy-efficiency and application QoS, and therefore must be efficiently controlled.

Congestion control generally follows three steps: congestion detection, congestion notification, and rate-adjusting. Congestion control protocol efficiency depends on how much it can achieve the following performance objectives: (i) First, energy-efficiency requires to be improved in order to extend system lifetime. Therefore congestion control protocols need to avoid or reduce packet loss due to buffer overflow, and remain lower control overhead that will consume additional energy more or less. (ii) Second, fairness needs to be observed so that each node can achieve fair throughput. Fairness can be achieved through rate-adjustment and packet scheduling (otherwise referred to as queue management) at each sensor node. (iii) Furthermore, support of traditional quality of service (QoS) metrics such as packet loss ratio and packet delay along with throughput may also be necessary.

There are several congestion control protocols [2]~[5] for sensor networks. They differ in the way that they detect congestion, broadcast congestion related information, and the way that they adjust traffic rate. CCF (Congestion Control and Fairness) [2] exactly adjusts traffic rate based on packer service time along with fair packet scheduling algorithms, while Fusion in [3] performs stop-and-start non-smooth rate adjustment to mitigate congestion. CODA [4] jointly uses end-to-end and hop-by-hop controls. Both CODA and ARC [5] employ AIMD-like (Additive Increase Multiplicative Decrease) coarse rate adjustment.

The existing congestion control protocols for WSNs only guarantee simple fairness, which means that the sink receives the same throughput from all nodes. In fact, sensor nodes might be either outfitted with different sensors or geographically deployed in different place and therefore they may have different importance or priority need to gain different throughput. Therefore weighted fairness is required to make sensor nodes get a throughput proportional to

Proceedings of the IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC'06)
0-7695-2553-9/06 $20.00 © 2006 IEEE

their priority. This paper investigates the problem of upstream congestion control in WSNs. We propose a new priority-based congestion control protocol (PCCP). Our contribution includes:

- We use packet inter-arrival time and packet service time in order to produce a measure of congestion. By incorporating information about packet inter-arrival time and the packet service time, we can capture congestion level at the node or at the link through a parameter, referred to as congestion degree, which is defined as the ratio of service time over inter-arrival time.
- PCCP realizes priority-dependent weighted fairness which allows sensor nodes to receive priority-dependent throughput. This model has not been considered by others.
- PCCP results in low buffer occupancy. As a result, it can avoid and/or reduce packet loss and therefore improve energy-efficiency. It achieves high link utilization and low packet delay.

The remainder of this paper is organized as follows: Section II describes system models considered in this paper. Section III presents PCCP in detail. Section IV provides simulation results. Finally, section V concludes the paper

2. System models

2.1. Network model

This paper addresses upstream congestion control for a WSN that supports single-path routing.

In Fig. 1, sensor nodes generate continuous data and form many-to-one convergent traffic in the upstream direction. They are assumed to implement CSMA-like MAC protocol. Each sensor node could have two types of traffic: source and transit. The former is locally generated at each sensor node, while the latter is from other nodes. Therefore each sensor node can be a source node and/or intermediate node. When a sensor node has offspring nodes and transit, it is a source node as well as an intermediate node. On the other hand, it is only a source node if it has no offspring nodes, and therefore only has source traffic. The offspring node of a particular node is defined as the node whose traffic is routed through this particular parent node. If an offspring node directly connects to its parent node, this offspring node is called child node and its parent node is called parent node. For example in Fig. 1, node A has 5 offspring nodes and therefore it plays the role of a source node as well as an intermediate node, simultaneously. Node C is the child node of node A, which in turn is the parent node of node C. However node B has zero offspring node and is only a source node. For a particular sensor node i, we use $O(i)$ to denote the total number of its offspring nodes. In the remainder of this paper, when we refer to sensor nodes, we mean that they act as both a source node and an intermediate node unless otherwise indicated.

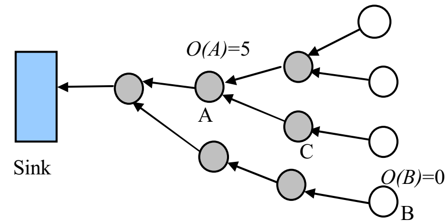


Figure 1. Network model - logical topology established by routing protocols.

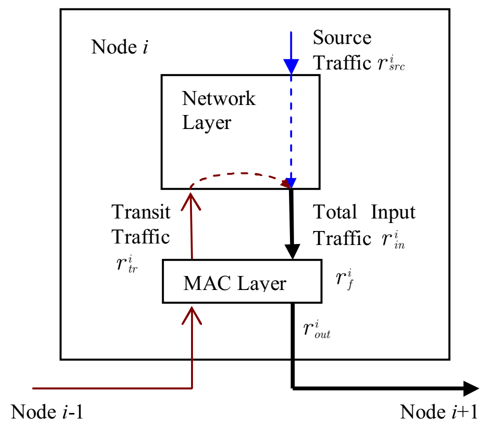


Figure 2. General node model.

2.2. Node model

Fig. 2 presents the queueing model at a particular sensor node i with single-path routing. The transit traffic of node i (r^i_{tr}) is received from its child nodes such as node $i-1$ through its MAC layer. The source traffic is locally generated with the rate of r^i_{src}. Both the transit traffic and the source traffic converge at the network layer before being forwarded to node $i+1$, which is the parent node of node i. Packets could be queued at the MAC layer if total input traffic rate ($r^i_{in} = r^i_{src} + r^i_{tr}$) exceeds packet forwarding rate at the MAC layer (r^i_f). The packet forwarding rate r^i_f depends on the MAC protocol itself. With the assumption of CSMA-like protocol, the number of active sensor nodes as well as their traffic density influences r^i_f. In Fig. 2, r^i_{out} is the packet rate at the node i towards node $i+1$. If r^i_{in} is smaller than r^i_f,

r_{out}^i will equal r_{in}^i. Otherwise if $r_{in}^i > r_f^i$, then r_{out}^i will be close to r_f^i. Therefore, $r_{out}^i = \min(r_{in}^i, r_f^i)$. This property can be utilized to indirectly reduce r_{out}^i through reducing r_{in}^i. In fact, the output traffic at node i is part of transit traffic at the node $i+1$. Therefore reduction of r_{out}^i implies a decrease of r_{tr}^{i+1}.

If packet input rate r_{in}^i exceeds packet forwarding rate r_f^i, then there will be backlogged packets inside node i and node-level congestion takes place. At this time, we need to reduce r_{in}^i and/or increase r_f^i. While r_f^i can be increased through adjusting MAC protocols, it is much easier to lower r_{in}^i through throttling either r_s^i, r_{tr}^i or both of them. The source rate r_{src}^i can be reduced locally by changing sampling (or reporting) frequency. The transit traffic r_{tr}^i can be indirectly reduced through rate adjustment at the node $i-1$. On the other hand, if there is collision on the link around the node i, then node i and its neighboring nodes should reduce channel access in order to prevent further link-level congestion. Although this task may be performed through MAC, yet it is easier to reduce r_{in}^i.

This paper designs a novel congestion control approach through flexible and distributed rate adjustment in each senor node as shown in Fig. 3. It introduces a scheduler between network layer and MAC layer, which maintains two queues: one for source traffic and another for transit traffic. The scheduling rate is denoted as r_{svc}^i. A Weighted Fair Queuing (WFQ) or Weighted Round Robin (WRR) algorithm can be used to guarantee fairness between source and transit traffic, as well as among all sensor nodes. The priority index of source traffic and transit traffic, which will be defined in next section, is used as the weight, respectively, for source traffic queue and transit traffic queue. By adjusting the scheduling rate r_{svc}^i, PCCP realizes an efficient congestion control while maintaining the MAC protocol parameters unchanged and therefore works well with any CSMA-like MAC protocol.

3. PCCP protocol

PCCP is designed with such motivations: 1) In WSNs, sensor nodes might have different priority due to their function or location. Therefore congestion control protocols need guarantee weighted fairness so that the sink can get different, but in a weighted fair way, throughput from sensor nodes. 2) Congestion control protocols need to improve energy-efficient and support traditional QoS in terms of packet delivery latency, throughput and packet loss ratio.

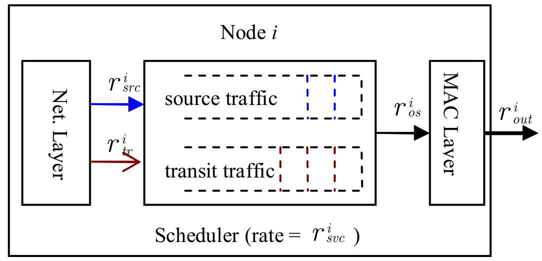


Figure 3. Node model in PCCP.

PCCP tries to avoid/reduce packet loss while guaranteeing weighted fairness and supporting multi-path routing with lower control overhead. PCCP consists of three components: intelligent congestion detection (ICD), implicit congestion notification (ICN), and priority-based rate adjustment (PRA).

ICD detects congestion based on packet inter-arrival time and packet service time. The joint participation of inter-arrival and service times reflect the current congestion level and therefore provide helpful and rich congestion information. To the best of our knowledge, jointly use of packet inter-arrival and packet service times as in ICD to measure congestion in WSNs has not been done in the past.

PCCP uses implicit congestion notification to avoid transmission of additional control messages and therefore help improve energy-efficiency. In ICN, congestion information is piggybacked in the header of data packets. Taking advantage of the broadcast nature of wireless channel, child nodes can capture such information when packets are forwarded by their parent nodes towards the sink.

Finally, PCCP designs a novel priority-base rate adjustment algorithm (PRA) employed in each sensor node in order to guarantee both flexible fairness and throughput, where each sensor node is given a priority index. PRA is designed to guarantee that: (1) The node with higher priority index gets more bandwidth; (2) The nodes with the same priority index get equal bandwidth. (3) A node with sufficient traffic gets more bandwidth than one that generates less traffic. The use of priority index provides PCCP with high flexibility in weighted fairness. For example, if the sink wants to receiver the same number of packets from each sensor node, the same priority index can be set for all nodes. On the other hand, if the sink wants to receive more detailed sensory data from a particular set of sensor nodes, such sensor nodes can be assigned a higher priority index and therefore allocated higher

0-7695-2553-9/06 $20.00 © 2006 IEEE

bandwidth. The following provides three definitions related to the priority index:

Definition 1: Source Traffic Priority ($SP(i)$) – The source traffic priority at sensor node i is used to represent the relative priority of local source traffic at node i. $SP(i)$ is independent of the offspring node number of the node i.

Definition 2: Transit Traffic Priority ($TP(i)$)– The transit traffic priority at sensor node i is used to represent the relative priority of transit traffic routed through node i. $TP(i)$ equals the sum of source traffic priority of each offspring node and depends on source traffic priority at all offspring nodes of node i. $TP(i)$ equals zero when node i has no offspring nodes.

Definition 3: Global Priority ($GP(i)$) – The global priority refers to the relative important of the total traffic at each node i. The global priority equals the sum of source traffic priority and transit traffic priority, or $GP(i) = SP(i) + TP(i)$. $GP(i)$ equals $SP(i)$ when node i has no offspring nodes.

3.1. Intelligent congestion detection (ICD)

In the traditional transport protocol such as TCP, congestion is often inferred at the end-points based on duplicated ACK messages or timer or ECN (Explicit Congestion Notification) [10] bit in the header of IP packets. In sensor networks, intermediate nodes participate in detecting congestion based on queue length [3], buffer increment [6], wireless channel status [7], or combination thereof. A single bit can be used to induce such information. The approach in [2] uses local packet service time to calculate sustainable service rate and in turn throttle node transmission rate. However these techniques cannot precisely reflect congestion level either at the node or at the link.

In order to precisely measure local congestion level at each intermediate node, we proposes intelligent congestion detection (ICD) that detects congestion based on mean packet inter-arrival (t_a^i) and mean packet service times (t_s^i) at the MAC layer. Here packet inter-arrival time is defined as the time interval between two sequential arriving packets from either source or for the transit traffic, and the packet service time is referred to as the time interval between when a packet arrives at the MAC layer and when its last bit is successfully transmitted. t_s^i covers packet waiting, collision resolution, and packet transmission times at the MAC layer. t_a^i as well as t_s^i can be easily measured at each node i on a packet-by-packet basis.

Based on the t_a^i and t_s^i, ICD defines a new congestion index, congestion degree $d(i)$, which is defined as the ratio of average packet service time over average packet inter-arrival time over a pre-specified time interval in each senor node i as follows:

$$d(i) = t_s^i / t_a^i . \qquad (1)$$

The congestion degree is intended to reflect the current congestion level at each sensor node. When the inter-arrival time is smaller than the service time, the congestion degree $d(i)$ is larger than 1 and the node experiences congestion. Otherwise when the congestion degree $d(i)$ is smaller than 1, the incoming rate is below the outgoing rate, and hence congestion abates. Therefore congestion degree can adequately represent congestion condition and provide helpful information in order to realize efficient congestion control. The congestion degree $d(i)$ can inform the child nodes about the traffic level to be increased or decreased by adjusting their transmission rate. In Eq. (1), t_a^i and t_s^i at each node i are measured using EWMA (exponential weighted moving average) algorithm.

In the process of determining the congestion degree, t_a^i is updated periodically whenever there are N_p (=50 in PCCP) new packets arriving as follows:

$$t_a^i = (1 - w_a) * t_a^i + w_a * T_{N_p} / N_p , \qquad (2)$$

where $0 < w_a < 1$ is a constant (= 0.1 in PCCP examples to be discussed later), T_{N_p} is the time interval over which the measurements are performed, and within which the N_p new packets arrive.

Also, t_s^i is updated each time a packet is forwarded as follows:

$$t_s = (1 - w_s) * t_s + w_s * t_s' , \qquad (3)$$

where $0 < w_s < 1$ is a constant (again 0.1 in the future examples), t_s' is the service time of the packet just transmitted.

3.2. Implicit Congestion Notification (ICN)

There are two approaches to propagate congestion information: Explicit Congestion Notification (ECN) and Implicit Congestion Notification (ICN). The explicit congestion notification uses special control messages and inevitably introduces additional overhead. In contrast, implicit congestion notification piggybacks congestion information in the header of data packets. Taking advantage of the broadcast nature of wireless channel, child nodes listen to their parent node to get congestion information. In the implicit congestion notification, transmission of an additional control message is avoided.

PCCP uses ICN at each sensor node i to piggyback congestion information in the header of data packets to be forwarded. Notification is triggered by either of the two events: (1) the number of forwarded packets by a node exceeds a threshold ($=O(i) * N_p$ in PCCP); (2) the node overhears a congestion notification from its parent node. The piggybacked information at a sensor node i includes mean packet service time (t_s^i), mean packet inter-arrival time (t_a^i), global priority $GP(i)$, and the number of offspring node $O(i)$. A node then computes its global priority index by summating its source traffic priority index and all the global priority index of its child nodes, which is piggybacked in the received data packets.

3.3. Priority-based Rate Adjustment (PRA)

As shown in Fig. 3, we introduce a scheduler with two sub-queues between the network layer and the MAC layer. If the scheduling rate r_{svc}^i is kept below the MAC forwarding rate r_f^i, the output rate will approximately equal the output rate r_{out}^i. Therefore, through adjusting the scheduling rate r_{svc}^i, congestion could be avoided or mitigated.

There are generally two ways to perform this task. The first is AIMD such as commonly used in the traditional TCP protocol or its variants. At this time, congestion information indicates whether there is congestion or not which can be transferred using a binary congestion notification (CN) bit. Unfortunately, AIMD is unable to exactly adjust the transmission rate because CN bit provides limited information. However when nodes are specifically informed as to how much to increase or decrease their rates, exact rate-adjustment becomes possible.

Congestion degree $d(i)$ and priority index ($TP(i)$ and $GP(i)$) introduced here provides more information than the CN bit and enables exact rate adjustment. PRA needs to adjust the scheduling rate r_{svc}^i and the source rate r_{src}^i at each sensor node after overhearing congestion notification from its parent node, in order to control both link-level congestion and node-level congestion.

01 $Initialization()$

02 $\quad d'(p_{i,0}) = 1, O'(p_{i,0}) = 0, r_{svc}^i = r_0;$

03 $SchRate(t_s^{p_{i,0}}, t_a^{p_{i,0}}, GP(p_{i,0}), O(p_{i,0}), r_{svc}^i, GP(i))$

04 $\quad d(p_{i,0}) = t_s^{p_{i,0}} / t_a^{p_{i,0}};$

05 $\quad total\_rate = 1 / t_s^{p_{i,0}};$

06 $\quad If(O(p_{i,0}) < O'(p_{i,0}))\ r_{svc}^i = r_{svc}^i / d(p_{i,0});$

07 $\quad If(O(p_{i,0}) > O'(p_{i,0}))\ r_{svc}^i = total\_rate * \frac{GP(i)}{GP(p_{i,0})};$

08 $\quad If(O(p_{i,0}) = O'(p_{i,0}))\quad \{$

09 $\qquad If(d(p_{i,0}) < d'(p_{i,0}))\ r_{svc}^i = r_{svc}^i / d(p_{i,0});$

10 $\qquad If(d(p_{i,0}) > d'(p_{i,0}))\ r_{svc}^i = base\_rate * \frac{GP(i)}{GP(p_{i,0})};$

11 $\quad \}$

12 $\quad d'(p_{i,0}) = d(p_{i,0}), O'(p_{i,0}) = O(p_{i,0});$

13 $\quad r_{svc}^i = \min(r_{svc}^i, 1 / t_s^i);$

14 $\quad return \quad r_{svc}^i * \eta;$

15 $SrcRate(r_{svc}^i)$

16 $\quad r_{src}^i = r_{svc}^i * \frac{TP(i)}{GP(i)};$

17 $\quad return \quad r_{src}^i;$

Figure 4. PRA in PCCP used by a node i to calculate its scheduling rate and source rate.

First we consider single-path routing, where each node i has only one parent node (let it be called $p_{i,0}$). Node i obtains the congestion information (including $t_s^{p_{i,0}}$, $t_a^{p_{i,0}}$, $GP(p_{i,0})$, and $O(p_{i,0})$) about node $p_{i,0}$ through the packets forwarded by $p_{i,0}$. Node i then updates its local scheduling rate and its source rate, respectively, using the procedures of SchRate() and SrcRate() as in Fig. 4. The initial scheduling rate is set at a small initial value r_0. We consider four cases in SchRate ():

1) First, when some offspring node becomes idle, packet inter-arrival time $t_a^{p_{i,0}}$ at node $p_{i,0}$ will increase and the congestion degree $d(p_{i,0})$ at the node $p_{i,0}$ will decrease, in this case, node i could scale-up its scheduling rate according to $d(p_{i,0})$ in order to improve link utilization (Line 6 of Fig. 4).
2) When new nodes become active, packet inter-arrival times $t_a^{p_{i,0}}$ will increase and therefore congestion degree $d(p_{i,0})$ will also increase.

$$r_{svc}^{i,j} = SchRate(t_s^{p_{i,j}}, t_a^{p_{i,j}}, GP(p_{i,j}), O(p_{i,j}), r_{svc}^{i,j}, GP(i) * w_j). \tag{4}$$

Node i may need to reduce its scheduling rate in this case. Keeping in mind the fact that all offspring nodes of node i are subset of offspring nodes of node $p_{i,0}$ because node i is a child node of node $p_{i,0}$, node i sets its scheduling rate to maximum allowable rate in order to guarantee fairness and high link utilization. The new scheduling rate is dependent on global priority index at both node i and its parent node $p_{i,0}$ (Line 7 of Fig. 4).

3) When the number of offspring nodes $O(p_{i,0})$ remains constant but some nodes don't have enough traffic, congestion degree $d(p_{i,0})$ will become smaller. So node $p_{i,0}$ can scale-up its scheduling rate according to $d(p_{i,0})$ in order to improve link utilization (Line 9 of Fig. 4).
4) In case Four, $O(p_{i,0})$ still remains the same but some nodes with small traffic, will produce more traffic. At this time, congestion degree $d(p_{i,0})$ will increase and possibly be larger than 1, and node i resets its scheduling rate back to the allowable rate as in case 2 so that to mitigate congestion while maintaining high link utilization (Line 10 of Fig. 4).

After scheduling rate is obtained, source rate of the node i (r^i_{src}) is updated based on its source traffic priority index $SP(i)$ and global priority index $GP(i)$ as shown in Line 15 of Fig. 4. In Line 14 of Fig. 4, η is a parameter smaller than but close to 1, which is used to maintain a small queue length and high throughput close to 1. In the future examples, η equals 0.98.

In the above rate adjustment process, global priority index is used to update the scheduling rate while the source traffic priority index is used to calculate source rate. Therefore, the rate adjustment has the following properties: (1) nodes with the same source traffic priority index get the same source rate; (2) nodes with a larger source priority index receive higher source rate and higher bandwidth.

4. Simulation results

This section presents the simulation results for PCCP. In simulations, we neglect the details of MAC protocols, but assume they provide even access opportunities for each node.

4.1. Simple Fairness

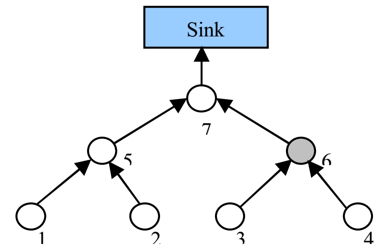


Figure 5. Simulation topology.

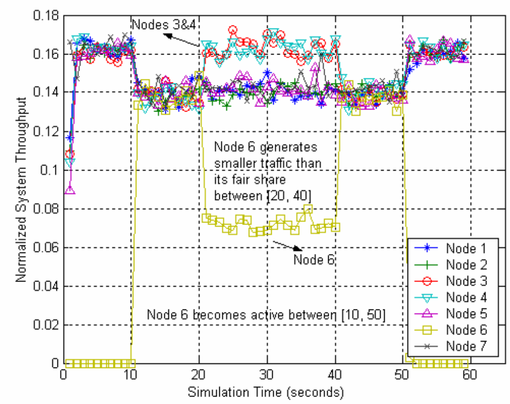
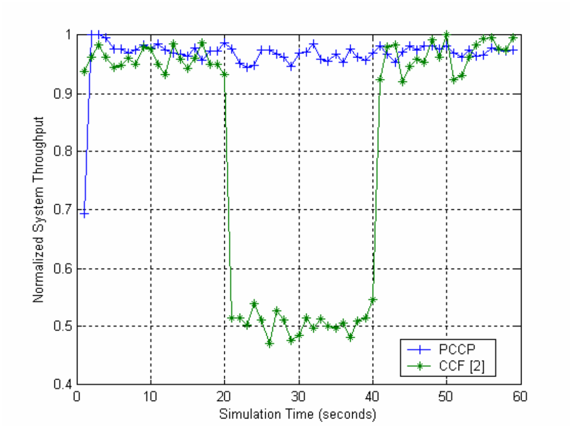


Figure 6. Normalized node throughput.

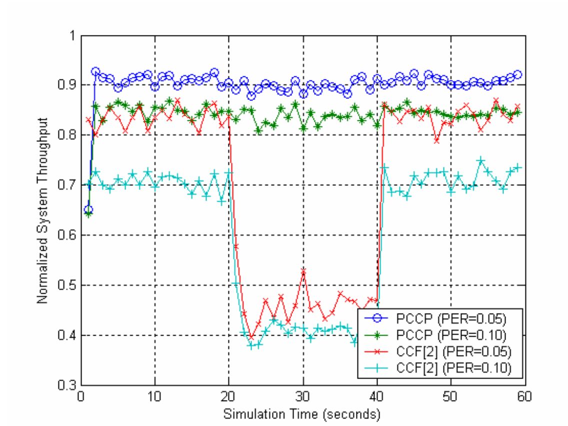
In this case, only single-path routing is assumed. We still assume that each sensor node has the same source traffic priority index and that the sink could obtain the number of packets from each sensor node. A simple tree topology as shown in Fig. 5 is assumed where there are 7 sensor nodes. Simulation time is 60 seconds. Sensor node 6 only remains active between [10 sec, 50sec] and generates a small traffic of 1/14 during [20sec, 40 sec], compared to the maximal normalized system throughput.. Other nodes are active throughout and have sufficient traffic throughout the simulation. The system bandwidth is normalized to 1, and therefore each node might receive a throughput of 1/7 in an ideal fair case. We assume that PER (Packet Error Rate) caused by bit error rate (BER) is zero.

Fig. 6 presents the normalized throughput for each node. From these results it can be observed that: (1) During the period [0.0 sec, 10 sec], the throughput of node 6 is zero since it remains idle during this interval, and other nodes receive nearly the same throughput; (2) After the 10 second interval, node 6 becomes active and all nodes starts to evenly share the total bandwidth. (3) During the interval [20 sec, 40 sec], node 6 generates small traffic (roughly 1/14), which is about half of its fair share. At this point, the output link of the node 6 is underloaded and its child nodes 3 as well as 4 increase their source rate and therefore receive a higher throughput. (4) During the interval [40 sec, 50 sec], node 6 returns to have enough traffic. Now,

0-7695-2553-9/06 $20.00 © 2006 IEEE



(a) PER = 0.0



(b) PER > 0

Figure 7. Normalized system throughput (PER = 0.0).

congestion degree at nodes 6 and 7 will increase and PCCP again lets each node receive the same throughput through the priority-based rate adjustment. (5) During the interval [50 sec, 60 sec], node 6 becomes idle and other nodes will receive nearly the same throughput.

We compare PCCP with CCF [2]. Fig. 7 (a) shows the sum of normalized throughput, respectively, for CCF and PCCP. Because CCF [2] cannot effectively allocate the remaining capacity and uses non-work-conservation scheduling algorithms, it has a lower throughput in the interval [20 sec, 40 sec] when node 6 does not have sufficient source traffic. When node 6 generates small traffic during the interval [20 sec, 40 sec], PCCP determines that packet inter-arrival time at the node 6 has increased and congestion degree at the node 6 is smaller than 1. Therefore child nodes of the node 6 (i.e., nodes 3 and 4), increase their source rate. As a result, PCCP maintains high throughput during this time interval as shown in Fig. 7 (a).

We next study the impact of PER on throughput of CCF and PCCP. The results are presented in Fig. 7 (b) where the PER are set at 0.05 and 0.1. When PER increases, packet inter-arrival time at each node will increase as well and the link is therefore underutilized. However, CCF only uses packet service time to detect congestion and therefore it cannot detect either underutilized links or nodes. As a result, its throughput is lower with an increase in PER. In PCCP, since it also relies on packet inter-arrival time to detect congestion, it is able to detect underutilized links and nodes resulting from PER. Therefore PCCP results in much higher throughput than CCF as shown in Fig. 7 (b).

4.2. Priority-based weighted fairness

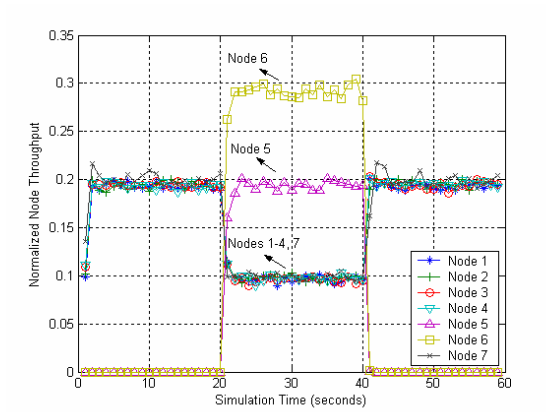


Figure 8. Weighted fairness: normalized node throughput.

In this case, we use the same topology as in Fig. 5, but the nodes will be configured with different source traffic priority index ($SP(i)$) as follows: node 6 with source traffic priority index 3, node 5 with source traffic priority index of 2, and all other nodes with source traffic priority index of 1. According to PCCP, node 5 will receive double the throughput and node 6 will obtain three times the throughput. Nodes 5 and 6 are set to become active during the interval [20 sec, 40 sec]. The result of normalized node throughput is shown in Fig. 8. During the interval [0 sec, 20 sec] or [40 sec, 60 sec], there are only 5 active nodes and PCCP allocated throughput of 0.2 to each active node. When nodes 5 and 6 become active in [20 sec, 40 sec], throughput of node 6 is around 0.3 while node 5 obtains a throughput of 0.2. Other nodes receive a throughput of 0.1. Total throughput is close to 1.0. Therefore PCCP effectively supports flexible weighted fairness through configuration of source traffic priority

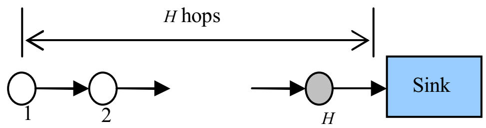


Figure 9. Linear topology.

index of each sensor node. If an applications desires to receive detailed data from sensor nodes in a special area, those nodes are given a higher priority index in order to let them receive higher bandwidth and report more frequently.

4.2. Impact of hop number

We also study the impact number of hops between sensor nodes and the sink on the performance of PCCP. The linear topology in Fig. 9 is used. Hop number (h) is varied between 10, 20, 30, and 40. We collect queue length at the node which is closest to the sink and calculate fairness index of each node's throughput according to the fairness definition given in [6]. As shown in Fig. 10, PCCP results in a short queue length even if $h = 40$ and it can avoid/reduce packet loss due to buffer overflow, which also results in improved energy efficiency. The reason the queue length decrease when h increases is the role of η in the priority-based rate adjustment (see Line 14 in Fig. 4). The fairness index presented in Fig. 11 shows that PCCP provides good fairness (close to 1), although the fairness index declines when h increases.

5. Conclusions

PCCP is a hop-by-hop upstream congestion control protocol for WSNs. It has the following properties: (1) uses packet inter-arrival and service times to accurately measure congestion at each sensor node; (2) introduces node priority index and realizes weighted fairness. Simulation results show that: (1) PCCP achieves high link utilization and flexible fairness; (2) PCCP achieves small buffer size; therefore it can avoid/reduce packet loss and therefore improve energy-efficiency, and provide lower delay.

6. Acknowledgement

The research was supported in part by grants from RGC under the contracts HKUST6204/03E HKUST6104/04E and HKUST6165/05E, a grant from NSFC/RGC under the contract N_HKUST605/02, grants from National NSF China under the contracts 60429202 and 60573115. The authors would like to express their gratitude to OPNET Technologies Inc. for providing software license to carry out the simulations of this research

7. References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: A survey," *Computer Networks*, vol. 38, 2002, pp. 393-422.

[2] C.-T. Ee and R. Bajcsy, "Congestion control and fairness for many-to-one routing in sensor networks," in *Proc. of ACM Sensys'04*.

[3] B. Hull, K. Jamieson, and H. Balakrishnan, "Mitigating congestion in wireless sensor networks," in *Proc. ACM Sensys'04*.

[4] C.-Y. Wan, S. B. Eisenman, and A. T. Campbell, "CODA: Congestion detection and avoidance in sensor networks," in *Proc. of ACM Sensys'03*.

[5] A. Woo and D. C. Culler, "A transmission control scheme for media access in sensor networks," in *Proc. of ACM Mobicom'01*.

[6] R. Jain, *The Art of Computer Systems Performance Analysis*, John Wiley and Sons, New York, 1991.

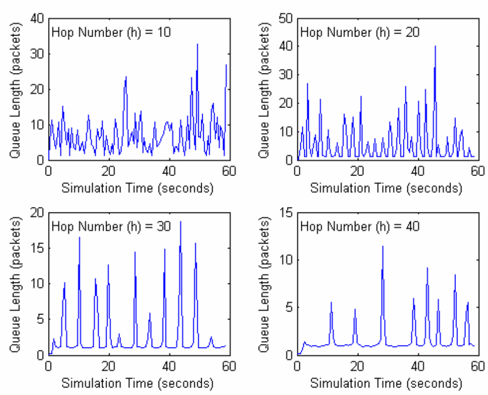


Figure 10. Queue length at the node closest to the sink.

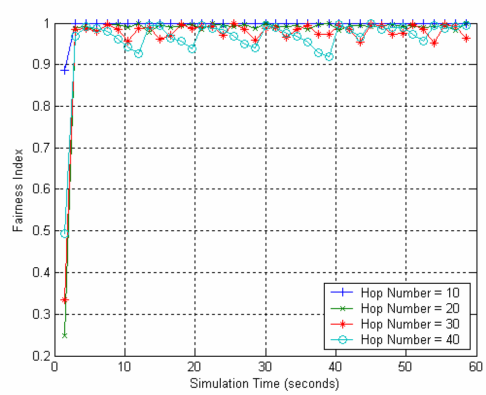


Figure 11. Fairness index.